Towards Semantic Software Engineering Environments

Ricardo A. Falbo, Giancarlo Guizzardi, Ana C.C. Natali
Gleidson Bertollo, Fabiano F. Ruy, Paula G. Mian

Computer Science Department, Federal University of Espírito Santo
Fernando Ferrari Avenue, CEP 29.060-900,
Vitória, Espírito Santo, Brasil
falbo@inf.ufes.br

ABSTRACT
Software tools processing partially common set of data should share an understanding of what these data mean. Since ontologies have been used to express formally a shared understanding of information, we argue that they are a way towards Semantic SEEs. In this paper we discuss an ontology-based approach to tool integration and present ODE, an ontology-based SEE.

Categories and Subject Descriptors
D.2.2 [**Software Engineering**]: Design Tools and Techniques – *Computer-aided software engineering (CASE)*.

General Terms
Design.

Keywords
Ontology, software engineering environments.

1. INTRODUCTION
As software process becomes more and more complex, it is necessary to provide computer-based tools to support software engineers to perform their tasks. To be effective, however, these tools must work together. Integration demands consistent representations of software engineering information, standardized interfaces between tools, homogeneous means of communication between software engineers and tools, and an effective approach that enables SEE to move among various platforms [1].

To deal with integration, we need an infrastructure. This infrastructure should be based on robust conceptual models. Software tools processing partially common set of data must share a common understanding of what they mean. Ontologies are a promising means to achieve these conceptual models, since they can serve as basis for comprehensive information representation and communication. In this way, an ontology-based approach can be used to improve tool integration in SEEs. Moreover, using such approach, we believe that we are going towards Semantic SEEs.

International Conference on Software Engineering and Knowledge Engineering, SEKE'02, July 15-19, 2002, Ischia, Italy.

2. ONTOLOGIES AND SEEs
Software Engineering Environments (SEEs) can be defined as integrated collections of tools that facilitate software engineering activities across the software lifecycle [2]. SEEs that explicitly establish a linkage between the tools and the software development process is called Process-centered Software Engineering Environments (PSEEs). A PSEE integrates tool support for software artifact development with support for the modeling and execution of the software process [2].

Because software processes are complex entities, a number of languages and modeling formalisms (often called Process Modeling Languages or PMLs) has been proposed. However, existing PMLs are complex, extremely sophisticated, and strongly oriented towards detailed modeling of processes. This occurs mainly because most software process researchers try to model all the details concerning software development. The problems with existing PMLs are reflected into PSEEs [3].

In this context, we explored the use of an ontology to improve process integration in a SEE. An ontology is a representation vocabulary specialized to some domain or subject matter. More precisely, it is not the vocabulary as such that qualifies as an ontology, but the conceptualizations that the terms in the vocabulary are intended to capture [4]. Ontologies are quintessentially content theories, because their main contribution is to identify specific classes of objects and relations that exist in some domain. Without ontologies, or the conceptualizations that underlie knowledge, there cannot be a vocabulary for representing knowledge [4]. An ontology should require only the minimal ontological commitment sufficient to support the intended knowledge sharing activities (minimal ontological commitment). It should make as few claims as possible about the world being modeled, allowing the parties committed to the ontology freedom to specialize and instantiate the ontology as needed [5].

Since an ontology does not intend to describe all the knowledge involved in a domain, but only that one that is essential to conceptualize the domain (minimal ontological commitment [5]), a software process ontology can be used as a coarse-grained process model that can be enriched when necessary. Moreover, an ontology can be developed without commitment to a specific formalism (minimal encoding bias [5]). Several approaches can be used to implement it using different technologies.

But this ontological approach for process integration can be generalized. In fact, we claim that if the tools in a SEE are built based on ontologies, tool integration can be improved. The same ontology can be used for building different tools supporting correlated software engineering activities. Moreover, if the

ontologies are integrated, integration of tools built based on them can be highly facilitated. Adopting this strategy, we believe that we can advance towards Semantic SEEs.

A Semantic SEE can be viewed as a SEE in which part of the information handled has a formal meaning (semantics) associated, augmenting its tools' ability to work in cooperation each other and with human developers. Tools committed them self with an ontology can share knowledge, since the ontology defines the common meaning.

The term "Semantic SEE" was coined using an analogy with Semantic Web [7]. Semantic Web aims to organize Web information, adding meaning to them, and allowing machines to process and analyze Web contents. The main goal of a Semantic SEE is analogous: to organize software engineering information, adding meaning to them, and allowing tools to share information. In a Semantic SEE, software engineering knowledge is accessible not only to human developers, but also to automated tools. Adapting the discourse of Bechhofer et al. [7] to our context, the key idea is to have software engineering data on the SEE defined and linked in such a way that its meaning is explicitly interpretable by software tools rather than just being implicitly interpretable by human developers.

Since ontologies is used to express formally a shared understanding of information, we advocate their use to go ahead Semantic SEEs. Ontologies can be developed to address software engineering sub-domains, such as software process, software quality, artifact modeling, and so on. Based on an ontology, domain infrastructures can be developed. Since we are most interested in the object technology, we have applyed an approach for deriving object frameworks from domain ontologies described in [6]. In this way, we can develop object-based domain infrastructures derived from the ontologies. These infrastructures, in turn, are used to build and integrate tools in the SEE.

3. ODE: AN ONTOLOGY-BASED SEE

We have experimented this ontological approach in ODE (**O**ntology-based software **D**evelopment **E**nvironment), a process-centered SEE. ODE's process kernel was built based on a software process ontology [6]. Tools for process definition and project tracking were also built based on this ontology.

To address software quality control in ODE, we adopted the same strategy. First, we developed a software quality ontology. From this ontology, we derived an object framework [8] and based on it we built a tool for quality management.

ODE's architectural style reflects its basis on ontologies. It has two levels. The base or application level concerns application classes, which model the objects that address some software engineering activity. The meta-level (or knowledge level) defines classes that describe knowledge about objects in the base level. Figure 1 shows these two levels concerning process integration and quality control.

The classes in the meta-level are derived directly from the ontologies. So, we can view the meta-level objects as items of an ontology instantiantion. The classes in the base level are also built based on the ontologies. The main classes and associations are derived from the ontology, preserving the same constraints as Knowledge's model. Also several classes in the base level have a corresponding Knowledge class in the Knowledge package. In this way, the meta-level can be used to describe base-level objects' characteristics. However, since an ontology does not intend to describe all the knowledge involved in a domain, but only that one that is essential to conceptualize the domain, new classes, associations, attributes and operations are defined to deal with specific design decisions made in the application level. In fact, the ontology is a general, common sense model, and then it does not contain all necessary modeling elements to treat applications' requirements.

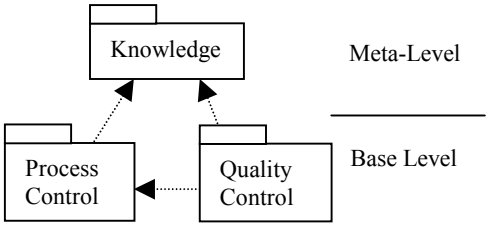

Figure 1 - ODE's two-layered architecture.

4. CONCLUSIONS

In this paper, we briefly discussed an ontological approach to deal with tool integration in SEEs and how it is materialized in ODE. Using such approach we argue that we are going towards what we are calling Semantic SEE: a SEE which tools are developed based on ontologies.

5. REFERENCES

[1] R.S. Pressman, *"Software Engineering: A Practitioner's Approach"*, 5th Edition, New York: McGraw-Hill, 2000.

[2] W. Harrison, H. Ossher, and P. Tarr, "Software Engineering Tools and Environments: A Roadmap", in Proc. of The Future of Software Engineering, ICSE, Ireland, 2000.

[3] A. Fuggetta, "Software Process: A Roadmap", in Proc. of The Future of Software Engineering, ICSE, Ireland, 2000.

[4] B. Chandrasekaran, John R. Josephson and V. Richard Benjamins, "What Are Ontologies, and Why Do We Need Them?", IEEE Intelligent Systems, January/February 1999.

[5] T. Gruber, "Towards principles for the design of ontologies used for knowledge sharing", Int. J. Human-Computer Studies, 43(5/6), 1995.

[6] G. Guizzardi, R. A. Falbo and J.G. Pereira Filho, "Using Objects and Patterns to Implement Domain Ontologies", in Proc. of the 15th Brazilian Symposium on Software Engineering, Rio de Janeiro, Brazil, 2001.

[7] S. Bechhofer, I. Horrocks, C. Goble, and R. Stevens, "OilEd: a Reason-able Ontology Editor for the Semantic Web", Proceedings of KI2001, Joint German/Austrian conference on Artificial Intelligence, Vienna. Springer-Verlag LNAI Vol. 2174, pp 396--408. 2001.

[8] R. A. Falbo, G. Guizzardi, and K. C. Duarte, "An Ontological Approach to Domain Engineering ", in Proc. of Int. Conference on Software Engineering and Knowledge Engineering, SEKE'2002, Ischia, Italy, 2002..